Rješavanje logičke zagonetke "mostovi" u programskom jeziku C

Rabuzin, Matija

Undergraduate thesis / Završni rad

2017

Degree Grantor / Ustanova koja je dodijelila akademski / stručni stupanj: **Josip Juraj Strossmayer University of Osijek, Faculty of Electrical Engineering, Computer Science and Information Technology Osijek / Sveučilište Josipa Jurja Strossmayera u Osijeku, Fakultet elektrotehnike, računarstva i informacijskih tehnologija Osijek**

Permanent link / Trajna poveznica: https://urn.nsk.hr/urn:nbn:hr:200:611762

Rights / Prava: In copyright / Zaštićeno autorskim pravom.

Download date / Datum preuzimanja: **2024-03-28**

Repository / Repozitorij:

Faculty of Electrical Engineering, Computer Science and Information Technology Osijek

zir.nsk.hr

Image not found or type unknown

SVEUČILIŠTE JOSIPA JURJA STROSSMAYERA U OSIJEKU
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA

Stručni studij

RJEŠAVANJE LOGIČKE ZAGONETKE „MOSTOVI“ U PROGRAMSKOM JEZIKU C

Završni rad

Matija Rabuzin

Osijek, 2017.

Sadržaj

1. UVOD 1
1.1 Zadatak završnog rada 1
2. PROGRAMSKI JEZIK C 2
2.1 Povijesni pregled razvoja programskih jezika 2
2.2 Razvoj C-a 3
2.3 C++ i njegova snovna svojstva 4
2.4 Usporedba s C-om 5
2.5 C# 6
3. MOSTOVI 7
3.1 Nastanak mostova 7
3.2 Pravila 8
3.3 Metode riješavanja 9
3.4 Rješavanje jedne zagonetke. 20
4. PROGRAMSKI DIO RADA 25
5. ZAKLJUČAK 29
SAŽETAK 30
ABSTRACT 31
LITERATURA 32
ŽIVOTOPIS 33

1. UVOD

U gotovo svim postojećim informacijskim sustavima može se primjetiti kako su građeni na temeljima algoritamskih znanja. Algoritmi imaju zadatak obraditi neprestano rastuće količine podataka i informacija koje karakteriziraju gotovo sve moderne sustave.

Zadatak algoritama i programera koji programiraju algoritam je da zadani problem riješe u što kraćem vremenskom roku, stoga kvaliteta pojedinca ovisi o njegovim sposobnostima i vještinama pisanja algoritama. Između prosječnog i kvalitetnog programera stoji znanje algoritama i struktura podataka koji stvaraju bitnu razliku. Programer mora imati u glavi program koji će raditi dobro, a ne onaj koji će raditi optimalno.

1.1 Zadatak završnog rada

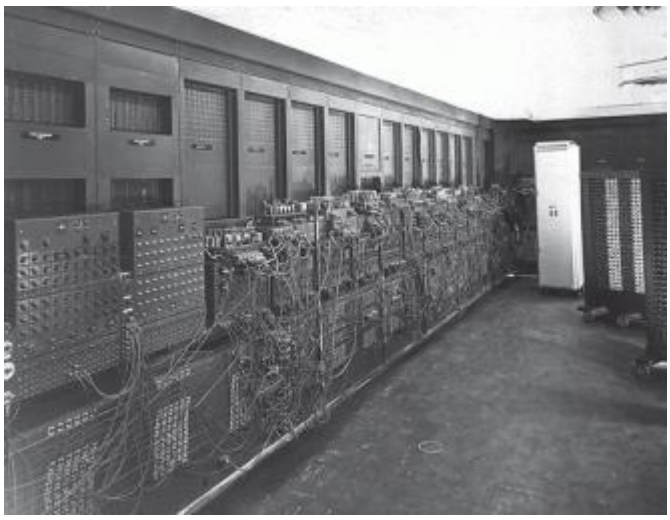
Zadatak završnog rada je napraviti algoritam koji će riješiti logičku zagonetku „Mostovi“ u programskom jeziku C. Treba napraviti program koji će uredno spojiti sve mostove iz zagonetke bez greške.

2. PROGRAMSKI JEZIK C

2.1 Povijesni pregled razvoja programskih jezika

Za jezik općenito možemo reći da je jedan od navažnijih ako ne i najvažniji proizvod ljudske civilizacije. Prirodni jezici su u svakodnevnoj uporabi kako bi se ljudi sporazumjevali i obavljali različite aktivnosti. Bogatstvo riječnika ovisi o podrijetlu jezika, starosti i kulture naroda iz kojeg taj prirodni jezik potječe, ali isto tako i o djelatnosti kojom se taj narod bavi. Veoma slična stvar je i sa programskim jezicima. Osmišljeni su kako bi obavljali različite aktivnosti i olakšali život ljudskoj civilizaciji. Isto tako i programske jezike možemo smatrati jednim od najvažnijih proizvoda ljudske civilizacije jer bez njih današnji svijet bi bio potpuno drugačiji, sav razvoj tehnologije možemo zahvaliti programskim jezicima jer bez njih moderne tehnologije ne bi bi bilo.

Kada su se prva računala pojavila bila su veoma komplicirana za korištenje. Takva računala su mogli koristiti isključivo stručnjaci koji su bili obučeni da mogu upravljati računalima i njihovim procesima. Prvi stručnjaci za računala, odnosno prvi programeri nisu imali veliki raspon mogućnosti rada na tim prvim računalima. Rad se sastojao u dva dijela. Prvi dio je bio da računalu daju upute koje računalo treba obaviti, dok je drugi dio bio da nakon toga iščitavaju rezultate dobivene nakon što računalo odradi upute koje su mu zadane. Prvotne upute su bile mnogo jednostavnije od današnjih uputa i programa koje današnji programski jezici mogu učiniti. Te upute su bile pisane u nizovima nula i jedinica davajući računala naredbe poput: „Zbroji dva broja“ , „Premjesti podatak s jedne memorijske lokacije na drugu memorijsku lokaciju“ i slično. Ovakve upute i zadatke bilo je veoma složeno za napisati te još složenije za pročitati i ispraviti eventualnu pogrešku stoga su nastali prvi programski alati nazvani assembleri (engl. Assemblers – sastavljači, to assemle –sastaviti).



Slika 1. - Primjer prvog računala

„U assemblerskom jeziku svaka strojna instrukcija predstavljena je mnemonikom koji je razumljiv ljudima koji čitaju program. Tako se zbrajanje najčešće obavlja mnemonikom ADD, dok se premještanje podataka obavlja mnemonikom MOV. Time se postigla bolja čitljivost programa, no i dalje je bilo vrlo složeno pisati programe i ispravljati ih jer je bilo potrebno davati sve, pa i najmanje upute računalu za svaku pojedinu operaciju.“[1]

Zaključak navedenog je da i dalje treba pronaći bolje rješenje koje će olakšati posao programerima da se mogu usredotočiti na sam bit problema. Ovo će kasnije dovesti do razvoja programskog jezika C i C++ .

Stoga ubrzo nakon toga razvijena su prva tri važnija programska jezika.

1. FORTRAN - programski jezik koji je bio u tadašnje doba izuzetno dobar za rješavanje matematičkih proračuna. Ime je dobio skraćivanjem dvije enleske riječi „**For**mula **Tran**slation“ . Razvijen je za računalo IBM 704 od strane tvrtke IBM između 1954. I 1957. godine.
2. BASIC – jednostavan program koji se veoma brzo mogao naučiti. Ime je dobio skraćenicom engleskih riječi „Beginner's All-purpose Symbolic Instruction Code“ što bi prevedeno na hrvatski značilo „Početnički svenamjenski simbolični instrukcijski kod“ . BASIC su osmislili 1964. godine John Kemeny i Thomas Kurtz te je bio iznimno popularan na računali 1980-ih godina
3. COBOL – program namijenjen upravljanju bazama podataka. Naziv je dobio skraćenicom engleskih riječi „Common Business Oriented Language“. Razvio ga je Short Range Comitee 1959. Godine.

2.2 Razvoj C-a

Nakon tih prvotnih programskih jezika dolazi se do razvitka prvog jezika opće namjene koji je postigao neviđeni globalni uspjeh, a to je programski jezik C. Programski jezika C je direktni predak današnjih jezika C++ te C#. Programski jezik C razvio je Dennis Ritchie 1970-ih godina u Bell Labs-u. „Jezik je bio jednostavan za učenje, omogućavao je modularno pisanje programa, sadržavao je samo narebe koje se mogu jednostavno prevesti u strojni jezik, davao je brzi kod. Jezik nije bio opterećen mnogim složenim funkcijama, kao na primjer skupljanje smeća(engl. Garbage collection) , ako je takav sustav nekome trebao korisnik ga je sam pisao“[2]. Programski jezik C tjesno je povezan sa UNIX operacijskim sistemom na kojemu je i razvijen. Pomoću C-a možemo odraditi velik broj uputa zadanih računalu kao što je:

1. Donošenje odluka (if-else),
2. Izbor(Switch),

[1] Julijan Šribar, Boris Motik, Demistificirani C++
[2] Julijan Šribar, Boris Motik, Demistificirani C++

3. Petlje s uvjetovanjem na početku(while),
4. Upućivanje da učini (Do),
5. Izlazi iz petlje (Break)

```
#include <iostream>

using namespace std;

int main()
{
    cout << "Hello world!" << endl;
    return 0;
}
```

Slika 2.- Najosnovniji program u C-u

Funkcije su vrlo bitan dio C programskog jezika. One mogu vraćati vrijednosti osnovnih tipova, struktura, unija ili pokazivača. Bilo koja funkcija može se rekurzivno pozvati. Funkcije se ne moraju nalaziti u istim izvornim datotekama već se mogu nalaziti u istim te se one posebno prevode. Varijable C programskog jezika mogu biti unutrašnje, vanjske i globalne.

2.3 C++ i njegova snovna svojstva

Programski jezik C++, proširenje i poboljšanje u odnosu na programski jezik C razvio je Bjarne Stroustrop. Bjarne se 1979. zaposlio u *Bell Labs* u Murray Hillu. Tamo je započeo raditi na svom projektu iz kojeg će se malo kasnije izroditi C++. Kao osnovu stvaranja C++ uzeo je jezik C.



Slika 3.- Bjarne Stroustrop

Postoje četiri najvažnija svojstva programskog jezika C++ koji ga čine objektno orijentiranim jezikom.

1. Učahurivanje (engl. encapsulation) – se odnosi na jedan od dva povezana, ali različita pojma, a ponekad i njihovu kombinaciju. Zabranjuje pristup nekim od komponenti objekta, te olakšava grupiranje podataka i fukncija koje operiraju nad tim podacima.
2. Skrivanje podataka (engl. Data hiding) – je mehanizam automatskog upravljanja memorijom. To znači da su okolini vidljivi samo oni podaci koji su potrebni za operiranjem nad pojedinim objektom. Tako svaki objekt ima samo svoje podatke koji nisu vidljivi odnosno ne smiju biti vidljivi drugim objektima.
3. Nasljeđivanje (engl. Inheritance) – ono omogućuje korištenje nekih članova i metoda osnovnog razreda od strane izvedenog razreda. Kao modifikator pristupa mogu se navesti ključne riječi public, private ili protected, koje tada mijenjaju pristup članovima objekta izvedenog razreda iz drugih dijelova programa.
4. Polimorfizam (engl. Polymorphism) – je kao što sama riječ kaže poprimanje više različitih obličja. U računalnom jeziku bi to značilo da metode mogu imati parametre različitih tipova ovisno o njihovoj potrebi, pa tako jedna metoda može imati i brojčane i tekstualne parametre. Polimorfizam također omogućuje da se adresa objekta podklase pohranjuje u pokazivač tipa njegove super klase.

Za razliku od starih programa koji su se pokretali sekvencijalno, većina današnjih programa se pokreće nekim događajem. Pomicanjem miša, klikom miša, pomoću prozora, izbornika, dijaloga i slično. Stoga C++ podržava programiranje pogonjeno događajem. To bi značilo da se program ne odvija po nekakvom unaprijed zadanom slijedu događanja već se odvija sukladno sa potrebama korisnika tog programa. Program tako postaje interaktivan te se pokretanjem nekog događaja pokreće taj dio programa te se naredba izvršava odmah te prikazuje rezultate na zaslonu računala.

Zaključak svega ovoga je to da je C++ veoma pogodan jezik opće namjene za izradu složenih programa. To mu omogućava njegovo jednostavno uvođenje novih tipova te naknadno dodavanje novih operacija.

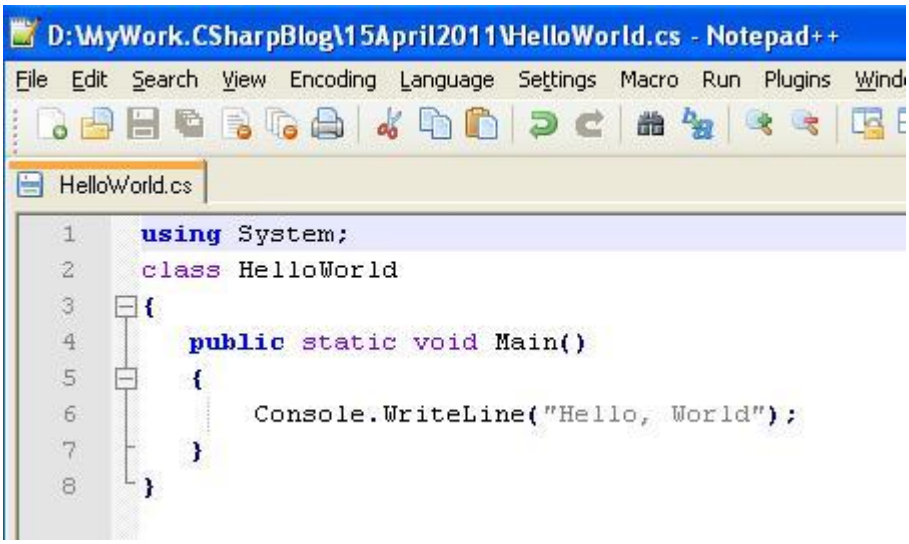
2.4 Usporedba s C-om

Na početku „života“ programskog jezika C++ mnogi su mu za veliku manu davali to što su programi pisani u C++ zahtjevali duže vrijeme obrade za razliku od C-a. Međutim kako je tržište raslo tako su i prevoditelji C++ jezika prerasli iz sporih prevoditelja u puno brže i efikasnije prevoditelje C++ jezika. No ipak, u pojedinim slučajevima izvedbeni kod dobiven iz C++ izvornog koda ipak može biti nešto sporiji u odnosu na onaj dobiven iz C-a. Stvar je na programeru da prosudi kada to postaje previše sporo i neefikasno da bi se tolerilalo.

Također mogućnost korištenje naslijeđivanja u programskom jeziku C++ ušteđuje programeru višestruko pisanje koda te olakšava ponovno korištenje već napisanih programskih odsječaka. Velika prednost C++ u odnosu na C je ta što je C++ puno jednostavniji za izradu programa. To će omogućiti jednostavnu izradu programa koje će biti lako za koristiti. Ono što C++ pruža bolje u odnosu na C je to što se može više pažnje posvetiti onome što želimo učiniti a ne jeziku i načinu na koji će se to odraditi.

2.5 C#

C# se može nazvati mlađim bratom C++ jezika. Njegov stvoritelj je Anders Hejlsberg i njegov tim iz tvrtke Microsoft. Isto kao i C++ jezik je objektno orijentiran te je opće primjene. C# namjenjen je izradu aplikacija .NET Framework platformu. Prva verzija najavljena je za 2000. godinu, no premijerno je objavljena 2002. godine pod imenom C# 1.0. Najnovija verzija C# 7.0 objavljena je u Ožujku 2017. godine.



Slika 4. - Najosnovniji program u C#

Iako C# i C++ imaju puno sličnosti postoje i određene razlike. „Jedna od najvažnijih promjena u odnosu na C++ je u tome da se zahtjeva mapa sa zaglavljima. Novi pristup je dobiven korištenjem tzv. folding editora, koji mogu "sklopiti i rasklopiti" kod metoda, odnosno da ga prikaže kao deklaraciju ili kao definiciju metoda." [3]Isto tako .NET runtime na kojemu se izvodi jezik C# sadrži upravljanje memorijom upotrebom objekata "garbage collector". Iz tog razloga je korištenje pointera u C# puno manje važne nego u C++ jeziku.

[3] Zoran Ćirović, Ivan Dunđerski, Tehnike vizualnog programiranja C#

Još neke razlike između C# i C++ su odsustvo globalnog prostora jer je sve u klasama, sve je izvedeno iz praklase Object, nema višestrukog naslijeđivanja, svaka je varijabla vrijednosti tipa za koji je varijabla deklarirana.

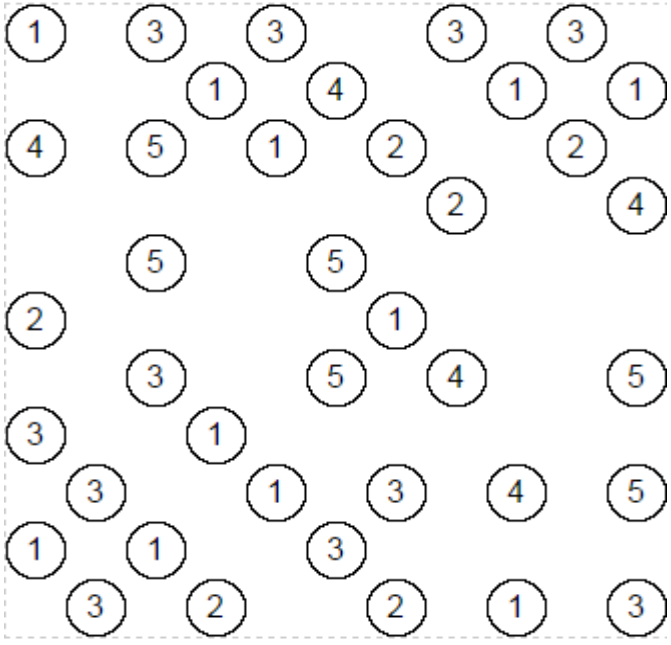
3. MOSTOVI

3.1 Nastanak mostova

Mostovi (jap. Hashiwokakero) je japanska logička igrica koju je prvi puta objavio časopis Nikoli. Časopis Nikoli je japanski časopis koji je najpoznatiji po svojim igricama, posebno po svojim logičkim zagonetkama. Nikoli je osnovan 1980. , a posebno se proslavio u svijetu svojom logičkom igricom Sudoku. Nikoli je objavio velik broj različitih logičkih zagonetaka. Neke najpoznatije njihove zagonetke su već spomenuti Sudoku, Kakuro, te Hashiwokakero.

U engleskoj je ta igrica bila objavljena pod imenom Bridges odnosno Chopsticks. U britanskom časopisu „The Times“ također se pojavila pod imenom Hashi. U francuskoj, danskoj, nizozemskoj i belgiji također je bila objavljivana ali pod imenom Ai-Ki-Ai.

Samo ime Hashiwokakero kada se doslovno prevede s japanskoga jezika znači „Sagradi mostove“. Mostovi se igraju na četvrtastom polju. Veličina polja nije fiksna, može biti raznih veličina ovisno težini same logičke zagonetke. Tako najčešće najmanja bude 7x7 za jako laganu razinu, pa sve do 20x20 za jako tešku razinu. Naravno one mogu biti i manje za recimo primjer rješavanja za nekoga tko se tek uči toj logičkoj igrici i mogu biti veće za one jako iskusne igrače koji žele što veći izazov.



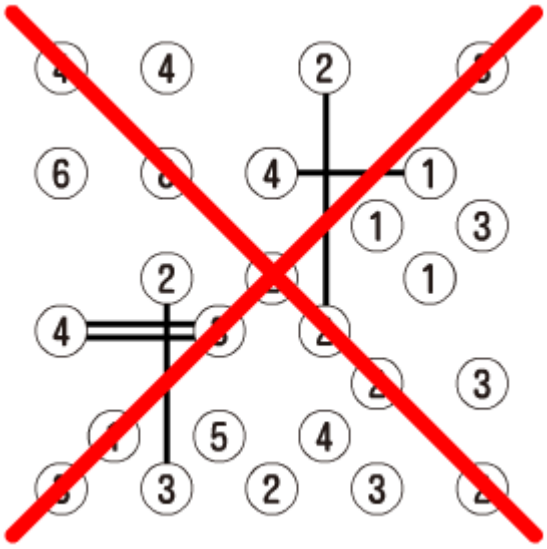
Slika 5. - Primjer jedne zagonetke „Mostovi"

3.2 Pravila

Na ploči postoje otoci s brojem unutar njih. Taj broj označava broj mostova koje se iz njih mogu povući. Brojevi sežu od 1-8.

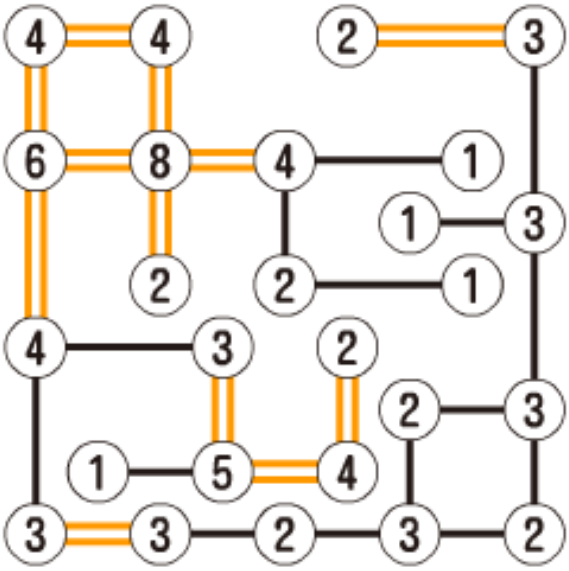
Pravila su veoma jednostavna ali se u okviru tih pravila mogu napraviti veoma teške slagalice. Pravila su sljedeća:

1. Most između dva otoka mora biti ravna neisprekidana crta, skretanja s putanje nisu moguća.
2. Križanja između mostova nisu moguća.



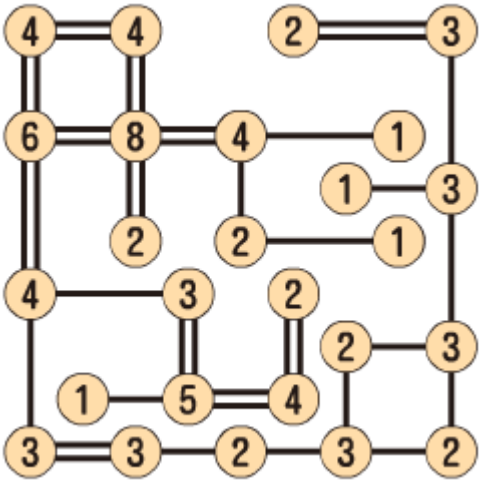
Slika 6. - Primjer drugog pravila

3. Mostovi se mogu kretati samo okomito i vodoravno, ne i dijagonalno.
4. Najviše sa dva mosta se može spojiti jedan par otoka.



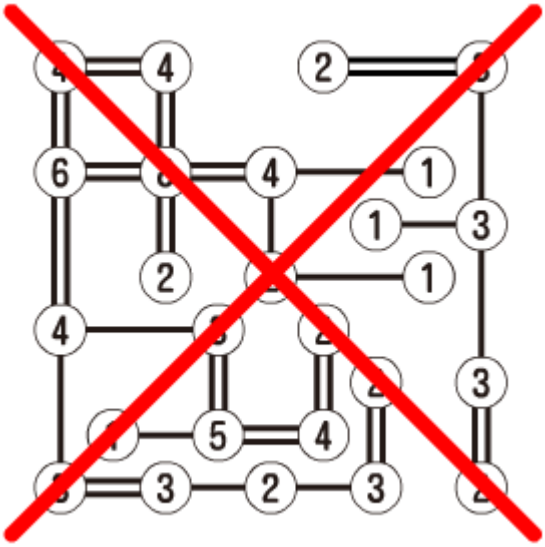
Slika 7. - Primjer četvrtog pravila

5. Broj mostova spojenih s jednog otoka mora odogovarati broju koji piše unutar tog otoka.



Slika 8. - Primjer petog pravila

6. Mostovi moraju biti spojeni u jednu jedinstvenu cijelinu, ne smiju se izolirati grupa otoka.

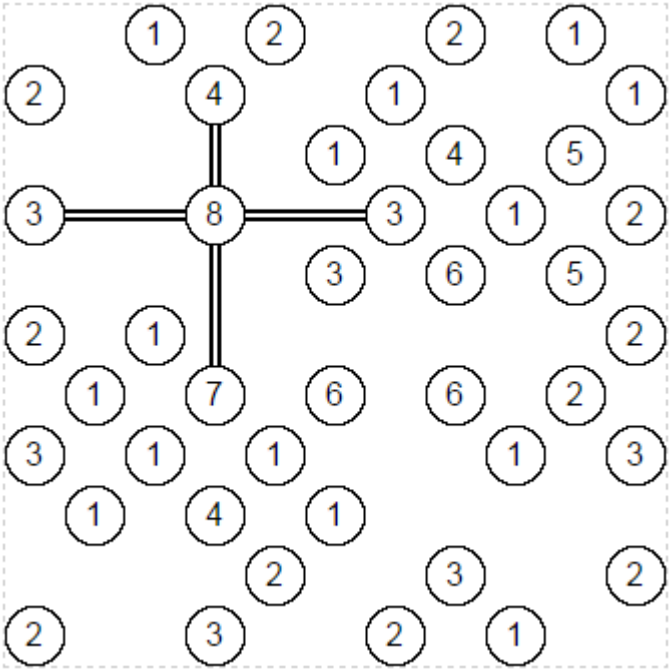


Slika 9. - Primjer šestog pravila

3.3 Metode riješavanja

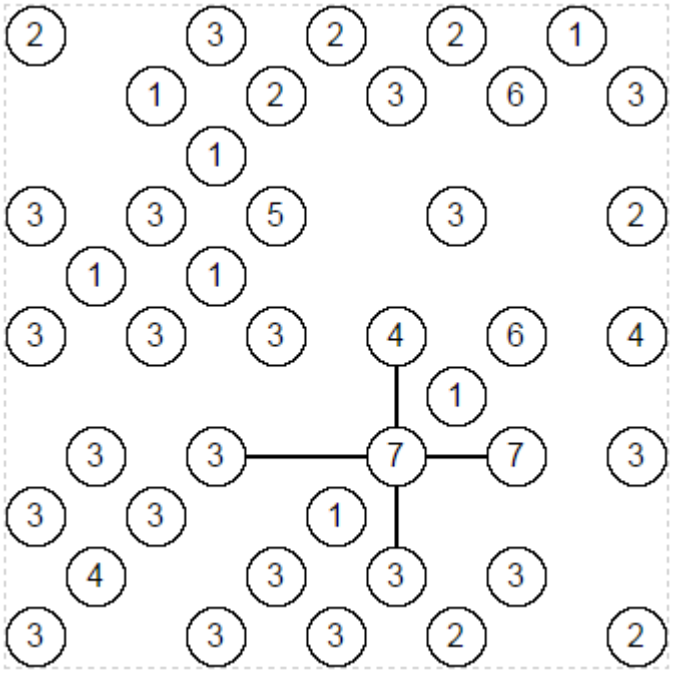
Najbolje je krenuti ako u slagalici postoji otok s brojem 8.

Otok s brojem 8 uvijek ima samo jedno jedino riješenje i uvijek je najbolje krenuti od toga. Otok s brojem 8 može povući 2 mosta gore, 2 mosta lijevo, 2 mosta dolje i 2 mosta desno. Iz razloga što se mostovi ne povlače dijagonalno ovo je jedino riješenje.



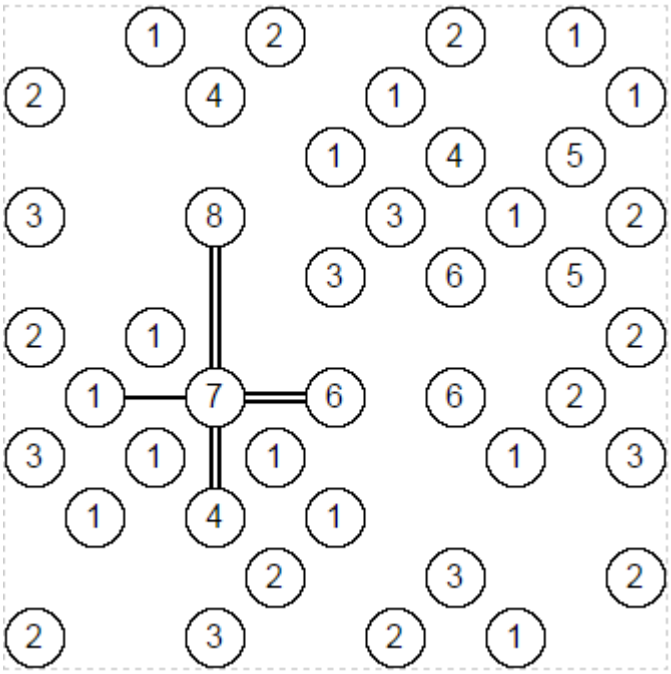
Slika 10. - Otok 8

Otok s brojem 7 može imati više rješenja. Prema 3 strane će povući dva mosta a prema jednoj strani jedan most. Iz ovoga se može izvući zaključak kako će prema svakoj strani povući barem 1 most pa je i to jedno sigurno rješenje za krenuti.



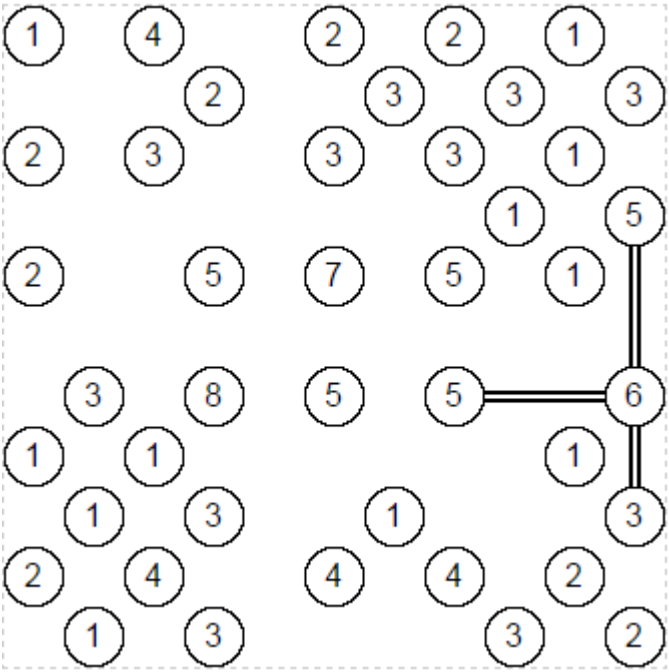
Slika 11. - Otok 7

Poseban slučaj broja 7 je taj kada na jednoj strani postoji otok s brojem 1 u sebi. U tom slučaju samo jedan most se može povući prema tom otoku, dok će se preostalih 6 mostova povući prema preostale 3 strane.



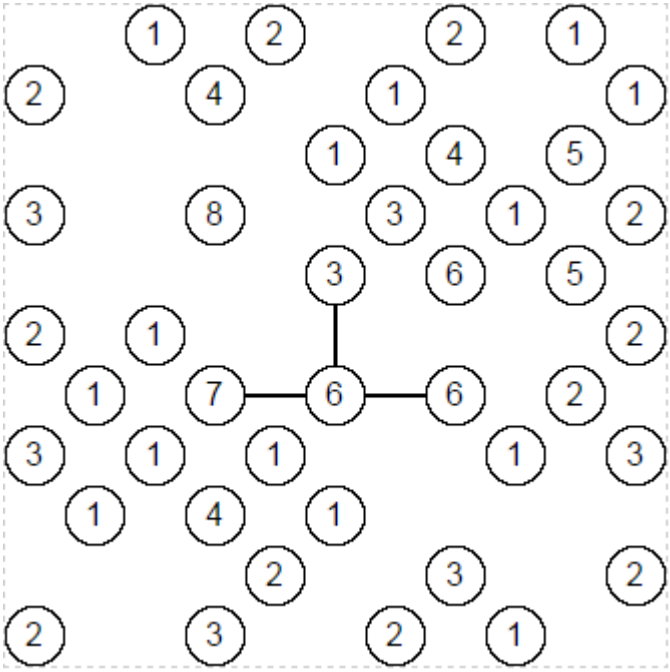
Slika 12. - Otok 7 s jednom stranom na kojoj je otok 1

Ako se otok s brojem 6 nalazi na rubu ploče tada on ima samo jedno moguće riješenje. Povući će po 2 mosta na 3 strane koje su mu preostale i to je jedino riješenje koje se u tom slučaju može napraviti.



Slika 13. - Otok 6 na rubu zagonetke

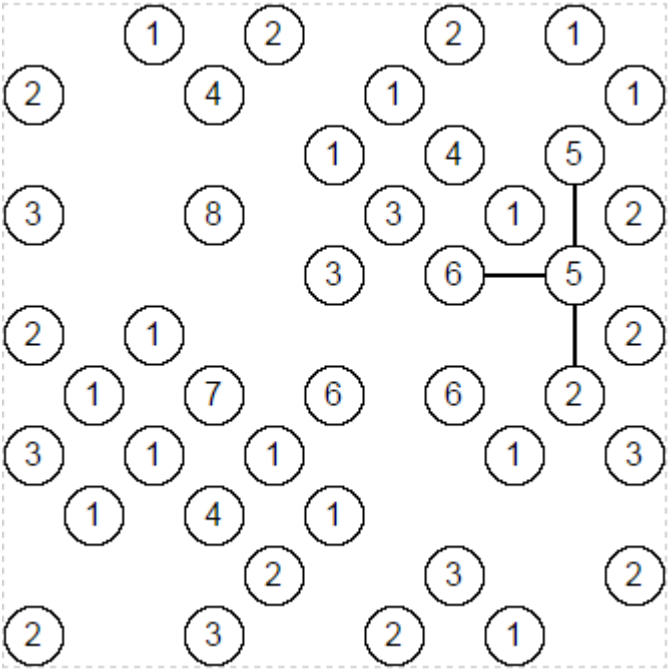
Posebni slučajevi kod otoka s brojem 6 je da ako se nalazi na ploči i može povući mostove na 4 strane a jedna ili dvije strane imaju otok s brojem 1. Ako je slučaj da ima jednu stranu gdje se nalazi otok s brojem 1, tada će sigurno na preostale tri strane povući barem 1 most.



Slika 14. - Otok 6 s jednom stranom na kojoj je otok 1

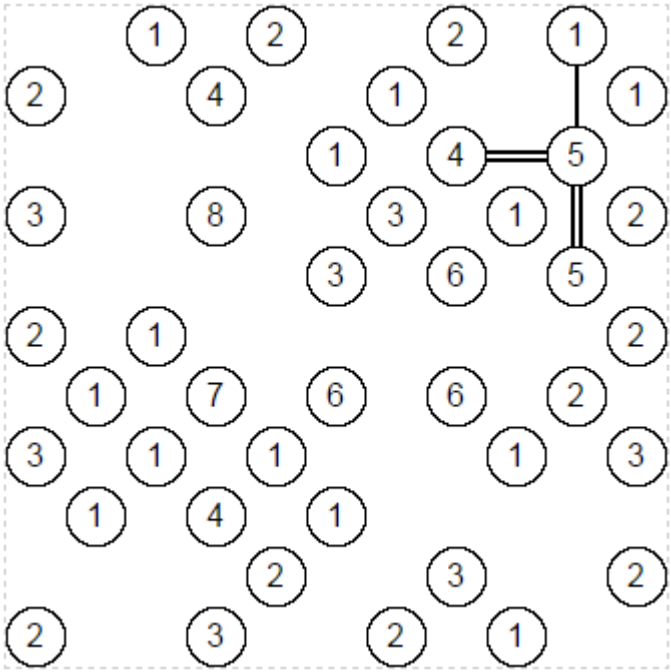
Isto tako ako ima dvije strane gdje se nalaze otoci s brojem 1. Tada će na preostale dvije sigurno morati povući po 2 mosta.

Ako se otok s brojem 5 nalazi na rubu ploče tada će sigurno na tri strane na koje može povući most, povući barem 1 most.



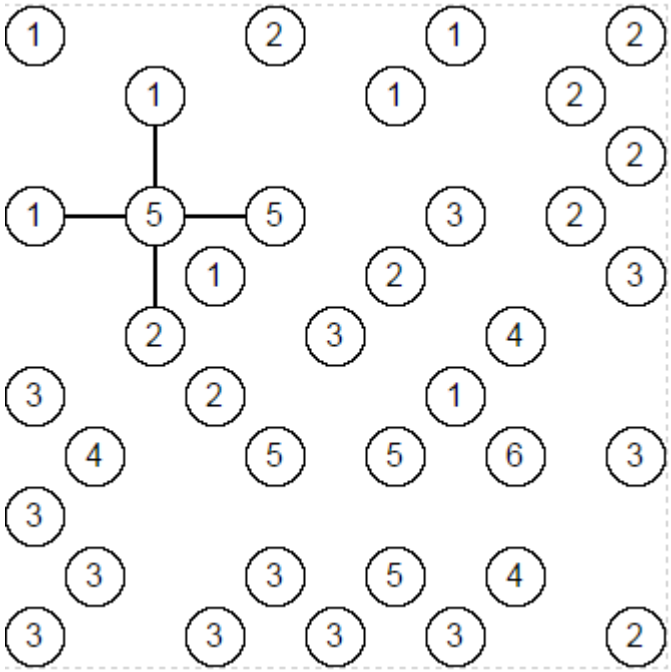
Slika 15. - otok 5 okružen s tri strane

Ako jedna od tih strana ima otok s brojem 1, tada će sigurno na preostale dvije strane povući dva mosta.



Slika 16. - Otok 5 okružen s tri strane, od toga je jedan otok 1

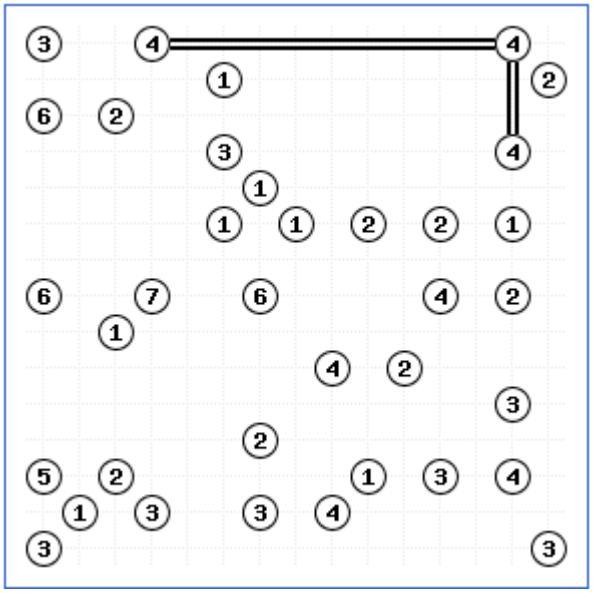
Posebni slučajevi kod otoka s brojem 5 je da ako se nalazi na ploči i može povući mostove na 4 strane a dvije ili tri strane imaju otok s brojem 1. Ako je slučaj da ima dvije strane gdje se nalazi otok s brojem 1, tada će sigurno na preostale dvije strane povući barem 1 most.



Slika 17. - Otok 5 okružen s četiri strane, od toga su dvije otok 1

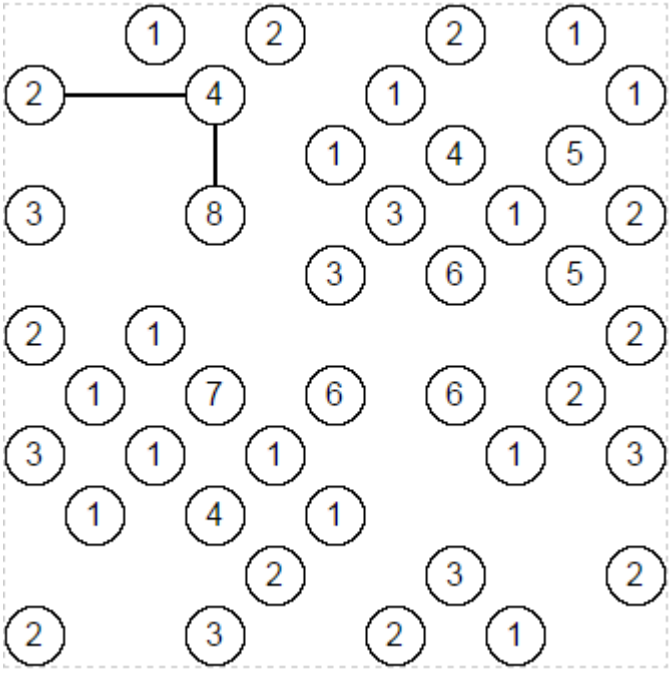
Isto tako ako ima tri strane gdje se nalaze otoci s brojem 1. Tada će na preostalu jednu sigurno morati povući po 2 mosta.

Ako se otok s brojem 4 nalazi u jednom od kuteva ploče tada ima samo jedno riješenje, a to je da povuče po 2 mosta na preostale 2 strane.



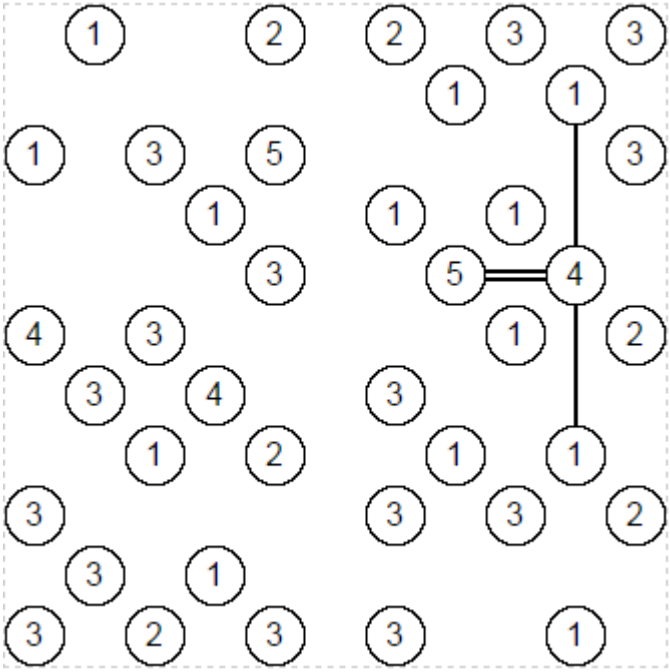
Slika 18. - Otok 4 okružen s dvije strane

Ako se nalazi negdje na rubu ploče ili negdje na ploči, a da može povući mostove na samo tri strane, te ako jedna ili dvije strane imaju otok s brojem 1 tada se može opet napraviti korak. Ako je slučaj da ima jednu stranu s brojem 1 tada će na preostale dvije sigurno povući barem 1 most.



Slika 19.- Otok 4 okružen s tri strane od kojih je jedna otok 1

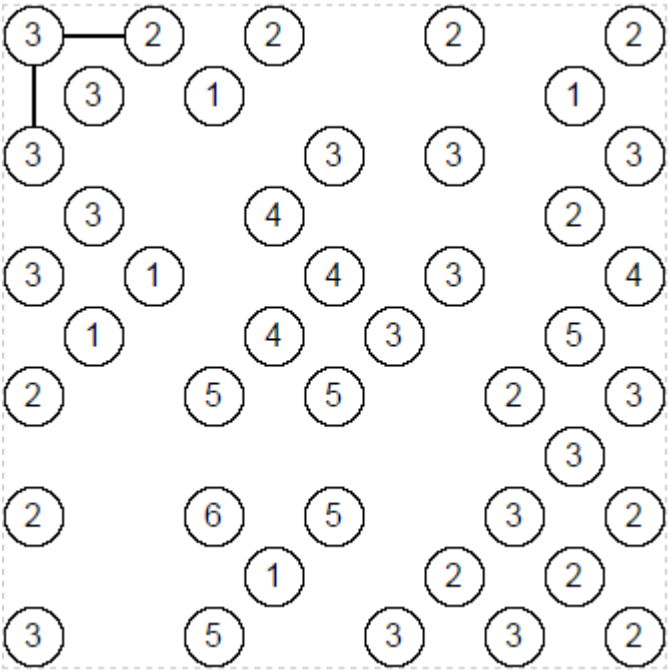
Ako je slučaj da ima dvije strane s otokom 1, tada će sigurno na onu jednu preostalu stranu povući dva mosta.



Slika 20. - Otok 4 okružen s tri strane od kojih su dvije otok 1

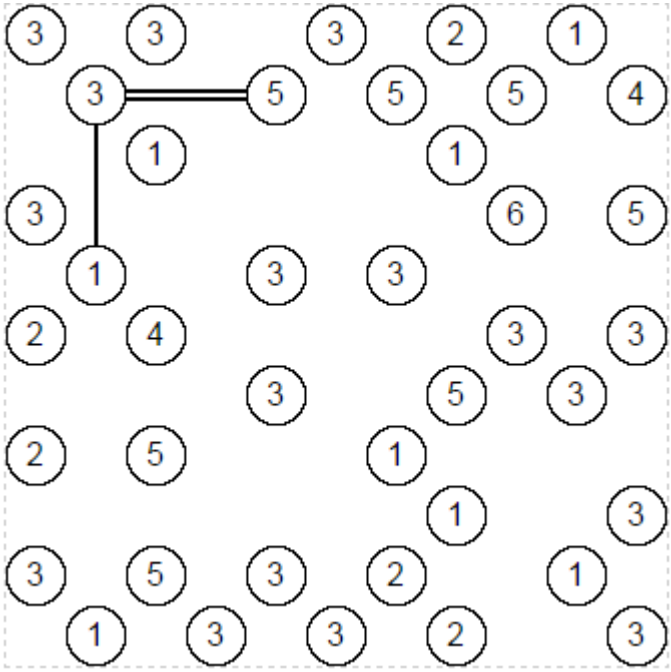
Ako se otok s brojem 4 nalazi negdje na polju da može povući mostove na četiri strane, a tri ili četiri strane imaju otoke s brojem 1 tada će sigurno povući po jedan most na svaku stranu.

Ako se otok s brojem 3 nalazi u kutu ploče ili negdje na polju da može povući mostove na samo dvije strane tada će sigurno prema svakoj strani povući barem jedan most.



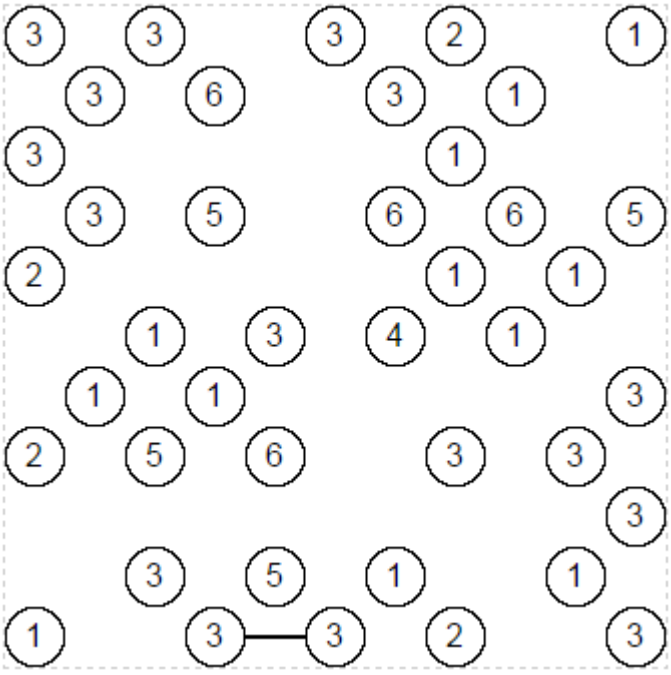
Slika 21. - Otok 3 u kutu zagonetke

Ako jedna od tih strana ima otok s brojem 1, tada će sigurno na onu drugu preostalu stranu povući 2 mosta.



Slika 22. - Otok tri okružen s dvije strane od kojih je jedna otok 1

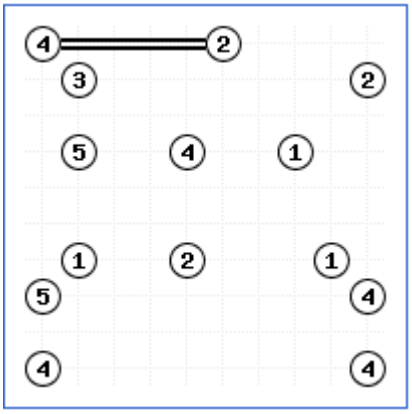
Ako se otok s brojem 3 nalazi negdje na polju gdje može povući mostove prema tri strane, a dvije sadrže otok s brojem 1. Tada će na onu treću stranu sigurno povući 1 most.



Slika 23. - Otok 3 koji je okružen s tri strane od kojih su dvije otok 1

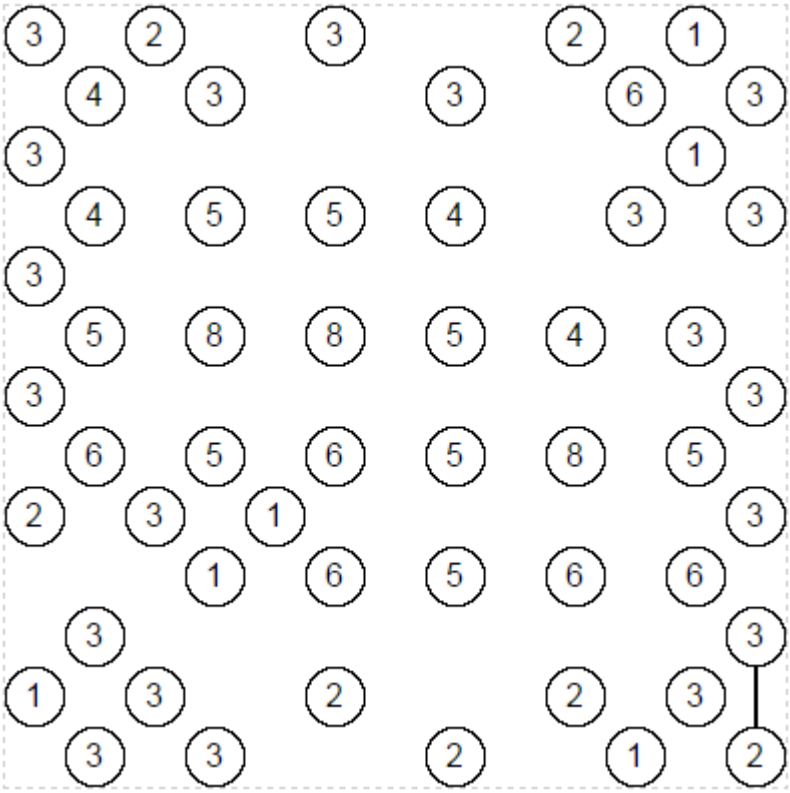
Ako se nalazi na negdje na polju gdje može povući mostove prema tri strane a sve tri strane sadrže otok jedan onda će na svaku stranu povući po 1 most.

Ako se otok s brojem 2 nalazi negdje na polju gdje može povući mostove samo prema jednoj strani, tada će to i napraviti jer je to jedino moguće riješenje.



Slika 24. - Otok 2 okružen samo jednom stranom

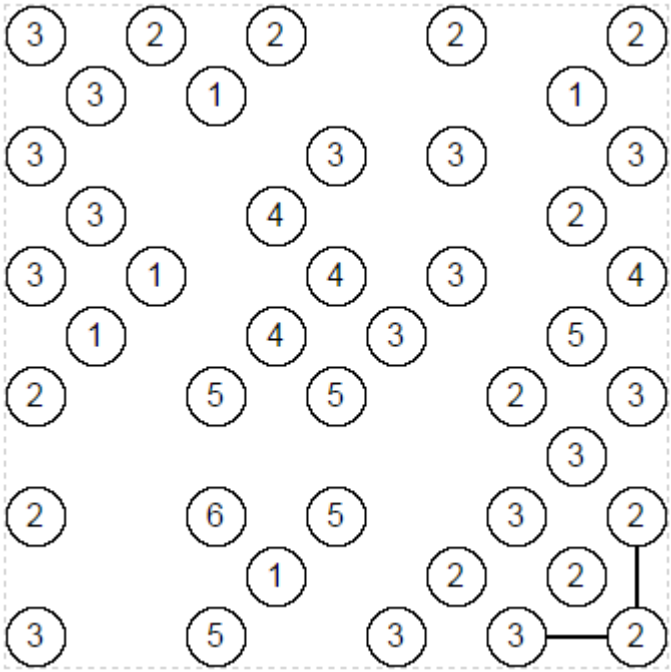
Ako se otok s brojem 2 nalazi negdje na polju gdje može povući mostove prema dvije strane, a jedna sadrži otok s brojem 1, tada će sigurno povući 1 most na onu drugu stranu.



Slika 25. - Otok 2 koji je okružen s dvije strane od kojih je jedna otok 1

Ako se pak nalazi negdje na polju gdje može povući mostove prema dvije strane, a obje sadrže otok s brojem 1, tada će sigurno povući 1 most na svaku stranu.

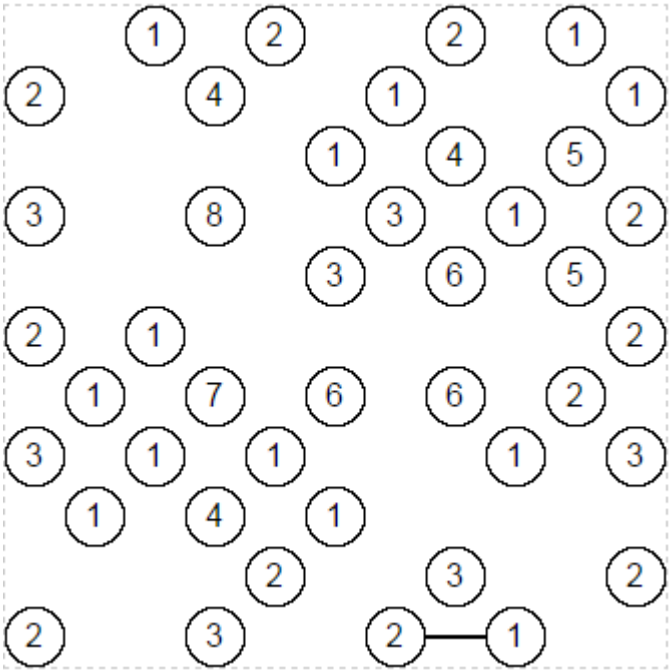
Ako se otok s brojem 2 nalazi negdje gdje ima dvije strane za povući most a na jednoj od tih strana je isto otok s brojem 2 tada će sigurno povući jedan most prema tom otoku a jedan most prema onome drugom jer prema pravilima, otoci moraju biti jedinstveno povezani, te grupa otoka ne smije biti izolirana.



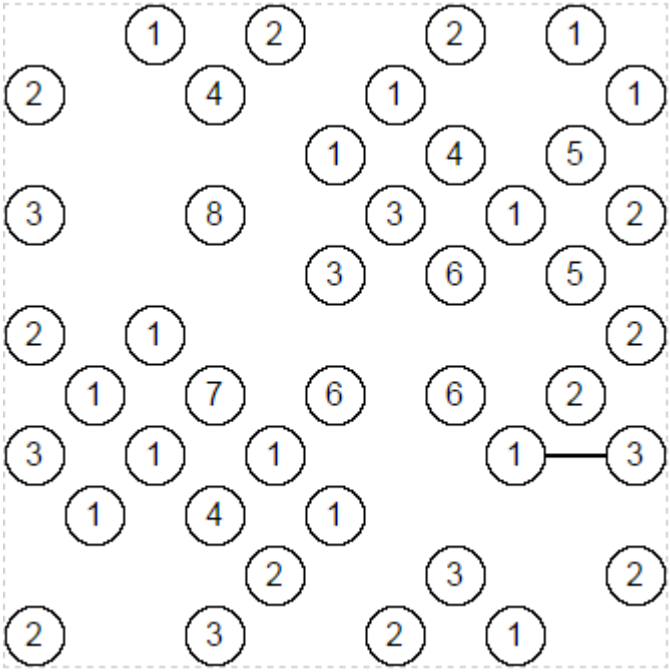
Slika 26. - Otok 2 okružen s dvije strane od kojih je jedna otok 2

Otok s brojem 1 se može odmah riješiti ako postoji samo jedna strana gdje može povući svoj most.

Još jedna solucija je kada ima dvije strane za povući mostove a jedna strana ima otok s brojem 1, kada ima tri strane za povući mostove a dvije sadrže otok s brojem 1, te kada ima četiri strane za povući mostove a tri strane sadrže otok s brojem 1, tada će sigurno povući most prema onoj strani koja nema otok s brojem 1, jer prema pravilima, otoci moraju biti jedinstveno povezani, te grupa otoka ne smije biti izolirana.

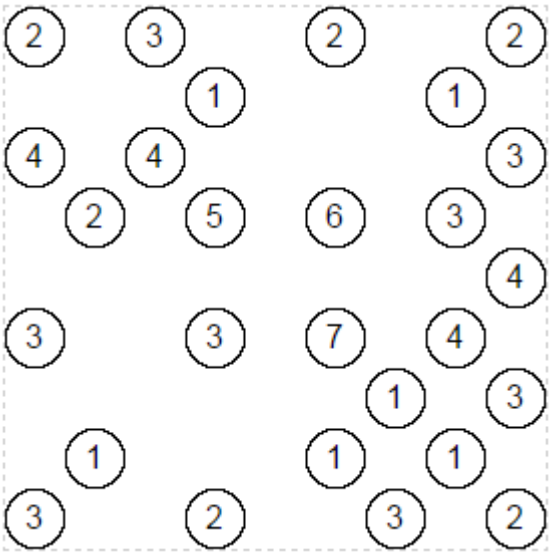


Slika 27. - Otok 1 okružen s dvije strane od kojih je jedna otok 1



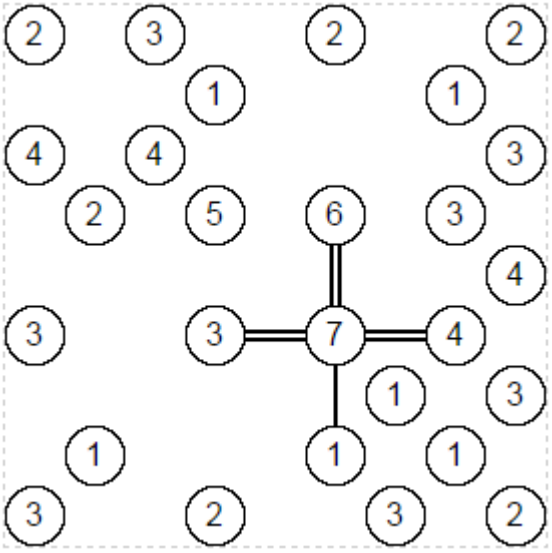
Slika 28. - Otok 1 okružen s tri strane od kojih su dvije otok 1

3.4 Rješavanje jedne zagonetke.



Slika 29. - Logička zagonetka mostovi

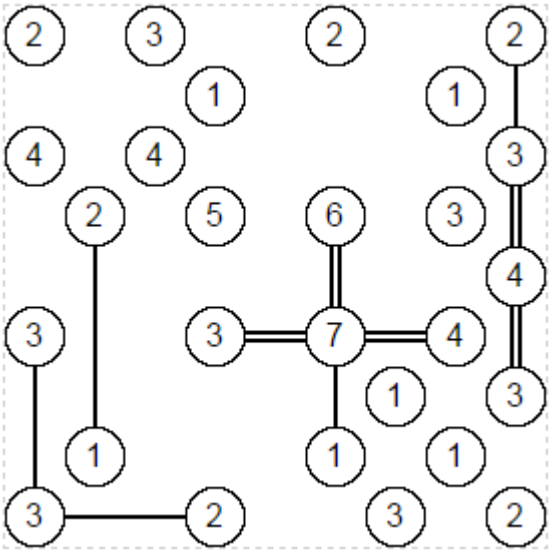
Prva stvar u ovoj zagonetki koja će se napraviti je da se riješi otok s brojem 7. Otok s brojem 7 u ovoj zagonetki ima susjedni otok s brojem 1 u njemu tako da se otok s brojem 7 može riješiti na samo jedan način.



Slika 30. - Prvi korak

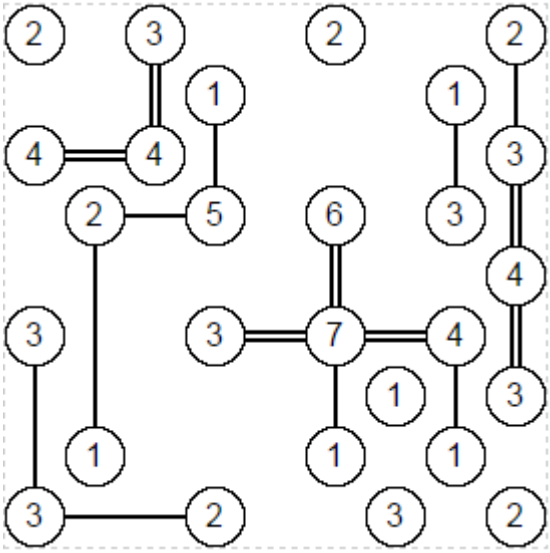
Sljedeći korak je promotriti kuteve slagalice. U gornjem desnom kutu imamo otok s brojem 2 koji za jedno od susjeda ima isto otok s brojem 2. U ovom slučaju će taj otok sigurno povući jedan most prema onom drugom susjedu. Isto tako u donjem lijevom kutu je otok s brojem 3. Onće sigurno povući po jedan most prema svakom susjedu. Odmah pored njega u drugom stupcu predzadnjeg reda imamo otok s brojem 1 koji ima dva susjeda a jedan od njih je isto

otok s brojem 1 tako da će on povući most prema onom preostalom susjedu. U zadnjem stupcu u petom redu je otok s brojem 4 koji ima samo dva susjeda te će prema svakome povući dva mosta.



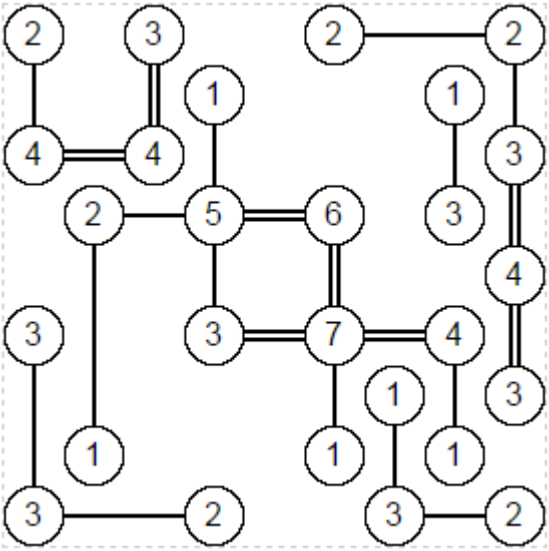
Slika 31. - Drugi korak

U trećem stupcu trećeg reda je otok s brojem 4 kojemu su preostala samo dva susjeda te će prema oba spojiti po dva mosta. U četvrtom stupcu drugog reda je otok s brojem 1 koji za jednog susjeda isto imam otok s brojem 1 tako da će on povući most prema onom drugom susjedu. U predzadnjem stupcu u drugom redu je isto otok s brojem 1 koji ima dva susjeda te je jedan od njih otok s brojem 1. On će isto tako povući most prema onom drugom susjedu. U predzadnjem stupcu predzadnje reda isto postoji otok s brojem 1 koji ima samo dva susjeda a jedan od njih je otok s brojem 1. I on će povući most prema onom drugom susjedu. U drugom stupcu u četvrtom redu je otok s brojem 2 koji treba povući još jedan most a ostao mu je samo jedan susjed kojemu to može napraviti.



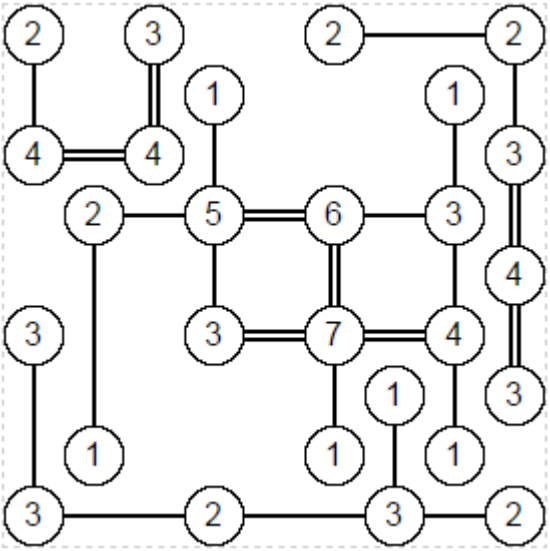
Slika 32. - Treći korak

Otok s brojem 2 u gornjem lijevom kutu ima dva susjeda. Od toga desni susjed je otok s brojem 3 koji može primiti još samo jedan most. To znači da će otok s brojem 2 sigurno povući barem 1 most prema donjem susjedu. U gornjem desnom kutu je isto otok s brojem 2 koji može povući još jedan most a preosala je samo jedna strana gdje to može napraviti. U četvrtom stupcu u četvrtom red je otok s brojem 5 kojem su preostala tri mosta za povući a može ih povući na dvije strane. Od toga donji susjed može primiti samo jedan most pa će otok s brojem 5 povući jedan most prema donjem susjedu a preostala dva prema desnom susjedu. Otok s brojem 1 u sedmom stupcu i sedmom redu može povući most samo prema jednoj strani. Otok s brojem 2 u donjem desnom kutu ima dva susjeda. Od toga gornji susjed može primiti samo jedan most pa će otok s brojem 2 sigurno povući jedan most lijevom susjedu.



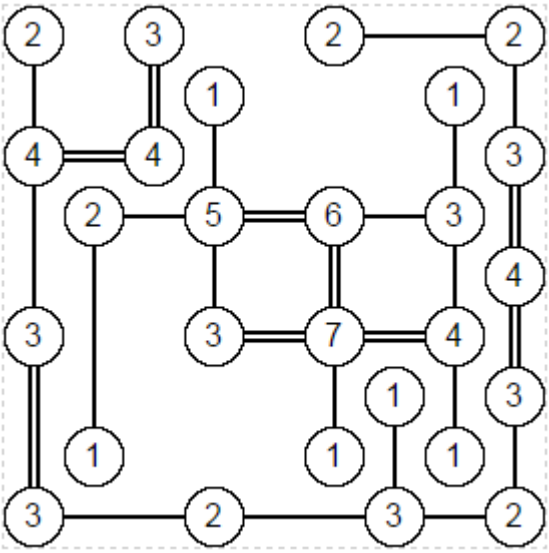
Slika 33. - Četvrti korak

U šestom stupcu i četvrtom redu nalazi se otok s brojem 6 kojem su preostala dva mosta za povući te dva susjeda kojima to može napraviti. Gornji susjed može primiti još samo jedan most stoga će otok s brojem 6 sigurno povući jedan most prema desnom susjedu. U osmom stupcu i šestom redu nalazi se otok s brojem 4 koji može povući još samo jedan most te ima samo jednu preostalu stranu gdje to može napraviti. U sedmom stupcu i zadnjem redu nalazi se otok s brojem 3. On može povući još jedan most a ima preostale dvije strane. Međutim ako most povuče prema desnom susjedu, otoku s brojem 2, napravit će izoliranu grupu od tri otoka koja neće biti spojena s ostatkom slagalice stoga to ne smije napraviti. Može tada povući most jedino prema svom lijevom susjedu.



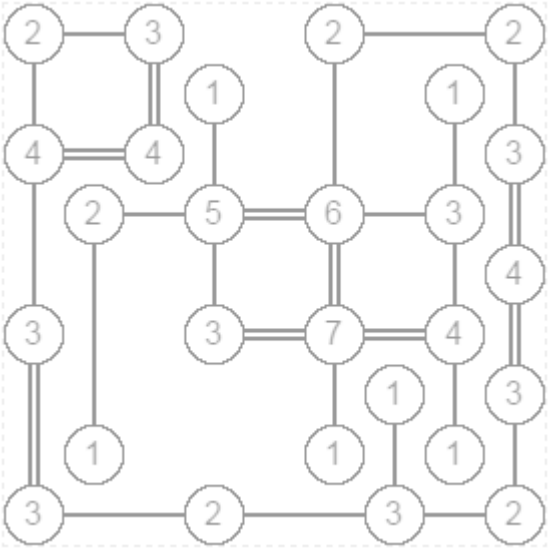
Slika 34. - Peti korak

U donjem desnom kutu nalazi se otok s brojem 2 koji može povući još jedan most te ima samo jednu stranu gdje to može napraviti. U donjem lijevom kutu nalazi se otok s brojem 3 koji može povući još jedan most te isto tako ima samo jednu stranu gdje to može napraviti. To će dovesti do toga da otok s brojem 3 koji se nalazi u prvom stupcu u šestom redu može povući još samo jedan most i to u gornju stranu.



Slika 35. -Šesti korak

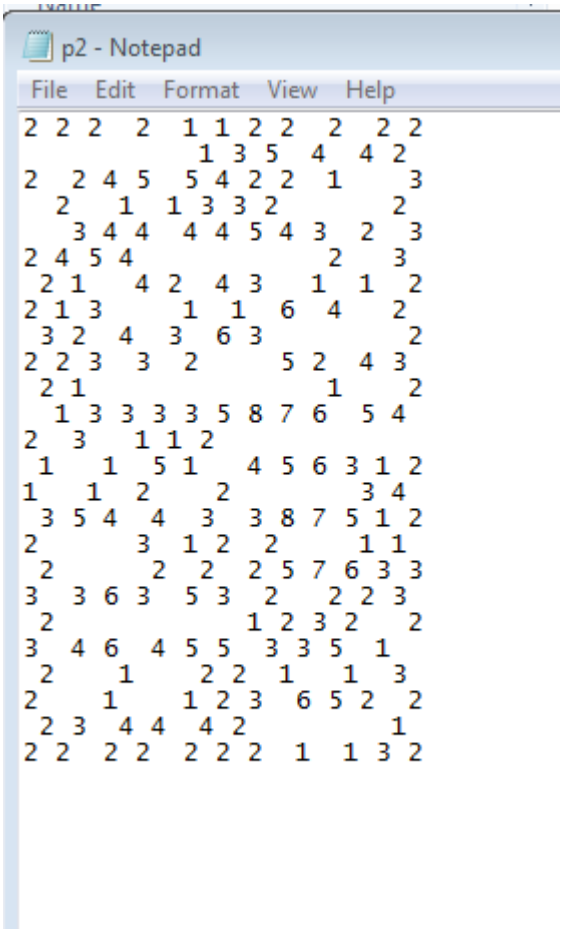
Ostalo je još samo spojiti otoke u prvom redu te otok s brojem 6. Otok s brojem 2 u gornjem lijevom kutu može povući još samo jedan most i to prema desnom susjedu, otoku s brojem 3. To će dovesti do toga da preostaje još samo jedan most za povući i to između otoka s brojem 6 i otoka s brojem 2 u prvom redu u šestom stupcu. Zagonetka je sada riješena.



Slika 36. - Šesti korak

4. PROGRAMSKI DIO RADA

Programski dio rada napisan je u C# jeziku. Stvoren je algoritam koji riješava sve zagonetke, počevši od onih najlakših, pa sve do onih najtežih. U program se može po želji ubaciti koja god se zagonetka želi kroz dva Notepad dokumenta u kojih se unese željena zagonetka.



Slika 37.- Primjer zagonetke unešene u Notepad dokument

Pokretanjem programa pokreće se izvršavanje algoritma te on u konzoli riješi zagonetku. U konzoli je to odrađeno na način da je zelenom bojom označeno kada je samo jedan most između dva otoka. Okomiti most napravljen je znakom „ | “ , dok je vodoravni most napravljen znakom „ - “. Crvenom bojom označeno je kada dva mosta povezuju otoke. Okomiti mostovi napravljeni su znakom „ H “, dok su vodoravni mostovi napravljeni znakom „ = “.



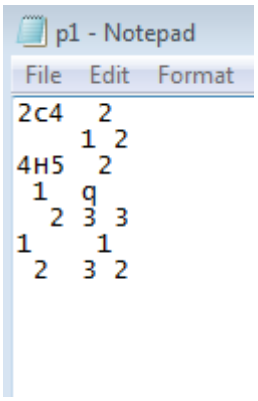
Slika 38.- Finalni program koji rješava zagonetku

Zbog testiranja postavljene su zagonetke i na razinu „Super hard“ . Ono što bi čovjeku predstavljalo pveći problem te bi mu trebalo duže da riješi takvu jednu zagonetku program riješava za manje od jedne sekunde.

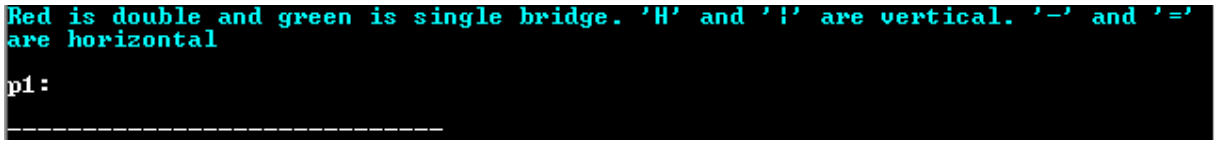


Slika 39.- Zagonetke postavljene na "Super hard" razinu

Ako se zagonetka unese u nepravilnom obliku s nepoznatim znakovima program ju neće riješiti, te to neće biti vidljivo kada se on pokrene.



Slika 40.- Nepravilno unesena zagrada



Slika 41.- Program neće niti pokušati riješiti nepravilno unesenu zagradu

5. ZAKLJUČAK

U ovom radu je napravljen algoritam za rješavanje logičke igrice mostovi u programskom jeziku C.

Zadatak ovog radio bio je prikazati kako se rješava logička zagonetka Mostovi te to prikazati u programskom jeziku C.

Kroz rad obuhvaćen je povjesni razvoj programskih jezika, te osnovne značajke programskih jezika C, C++ te C#.

Također u radu je prikazana logička zagonetka mostovi. Prikazan je njezin povjesni nastanak, njezin opis, pravila igre te metode i načini rješavanja.

SAŽETAK

Logička zagonetka Mostovi u programskom jeziku C.

Zadatak ovog rada bio je napraviti algoritam u programskom jeziku C koji će riješiti logičku zagonetku mostovi. U prvom dijelu rada napravljen je kratak osvrt na povijest i način rada programskog jezika C, C++ te C#. U nastavku opisana je logička igrica mostovi, njezina povijest, pravila te načini rješavanja. Predstavljeni su primjeri rješavanja zagonetke, te je jedna zagonetka u potpunosti riješena. U programskom dijelu rada opisan je način kako algoritam radi te je taj dio popraćen slikama izvedbe.

ABSTRACT

Logic puzzle „Bridges“ in programming language C.

The aim of this paper is to develop an algorithm of logic puzzle 'Bridges' in programming language C which will solve the puzzle. First part of paper starts with brief summary of history and mode of operation of programming language C, C++ and C#. This summary was followed by history, rules and ways of solving logic puzzle Bridges. In program development part of paper, a mode of operation of algorithm is described along with pictures of program execution.

LITERATURA

[1] Julijan Šribar, Boris Motik, Demistificirani C++

[2] Domagoj Kusalić, Napredno programiranje i algoritmi u C-u i C++-u

[3] Zoran Ćirović, Ivan Dunđerski, Tehnike vizualnog programiranja C#

[4] https://en.wikipedia.org/wiki/Hashiwokakero (vrijeme pristupa: 10.6.2017.)

[5] https://en.wikipedia.org/wiki/Nikoli_(publisher) (vrijeme pristupa: 10.6.2017.)

[6] http://www.hashi.info/ (opis, pravila i primjeri igre) (vrijeme pristupa: 10.6.2017.)

[7] http://www.nikoli.com/en/puzzles/hashiwokakero/rule.html (opis, pravila i primjeri igre) (vrijeme pristupa: 10.6.2017.)

[8] http://www.puzzle-bridges.com/ (primjeri igre)(vrijeme pristupa: 10.6.2017.)

ŽIVOTOPIS

Matija Rabuzin rođen je 2.8.1994 u Zagrebu. Dolazi iz Pakraca gdje je završio Osnovnu Školu Braće Radića. Paralelno s osnovnom školom upisuje se u Rukometni Klup LiPa 2006. Godine u kojemu trenira sve do 2014. godine. Nakon završene osnovne škole upisuje Opću Gimnaziju u Srednjoj Školi Pakrac. Po završetku srednje škole upisuje Elektrotehnički fakultet u Osijeku. Tijekom obrazovanja na fakultetu odradio je praksu u O.R.K.A-i(Osječkoj radionici kvalitetnih aplikacija). Također je preko studentskog servisa radio na dva posla. U Meggleu te u Transcomu.

Potpis: